第一章　　2023.03.16　　张蓝心

一、1、管理计算机软件和硬件的软件系统

4、指系统能及时响应外部事件的请求，在规定的时间内完成对该事件的处理，并控制所有实时任务协调一致地运行

5、当进程A要访问某资源时，必须先提出请求，若此时该资源空闲，系统便可将其分配给A使用。此后若再有其他进程也要访问该资源，则只要A未用完，其他进程就必须等待。仅当A访问完并释放后，才允许另一进程对该资源进行访问。

三、1、硬件系统　软件系统

3、硬件资源　软件资源

5、共享性　异步性

第二章

一、3、指在多进程/多线程环境下，访问共享资源的代码段

4、指协调多个并发进程/线程的执行顺序，确保他们正确有序地访问共享资源

三、1、动态性　异步性　制约性

5、动态　静态

6、间接制约/互斥制约

8、内存共享　消息传递　信号

4、运行　就绪　阻塞

五、2、~~S1是起始进程，无前趋~~

S2 S1→S2，S1→S3，S2→S4，S2→S5，S3→S6，S4→S7，S5→S7，S6→S7

```
semaphore s1_done = 0;
semaphore s2_done = 0;
semaphore s3_done = 0;
semaphore s4_done = 0;
semaphore s5_done = 0;
semaphore s6_done = 0;
semaphore s7_done = 0;
void S1( ){
    signal (s1_done);
}
void S2( ){
    wait (s1_done);
    signal (s2_done);
}
void S3( ){
    wait (s1_done);
    signal (s3_done);
}
```

```
void S4( ){
    wait (s2_done);
    signal (s4_done);
}
void S5( ){
    wait (s2_done);
    signal (s5_done);
}
void S6( ){
    ~~wait (s4_done);~~
    ~~wait (s5_done);~~
    wait (s3_done);
    signal (s6_done);
}
void S7( ){
    wait (s4_done);
    wait (s5_done);
    wait (s6_done);
}
```

3.
```
semaphore mutex = 1;
semaphore empty = 5;
semaphore apple = 0;
semaphore orange = 0;
void father( ){
  while (true){
    fruit = random (apple, orange);
    wait (empty);
    wait (mutex);
    plate.add (fruit);
    if (fruit == apple){
      signal (apple);
    }
    else{
      signal (orange);
    }
    signal (mutex);
  }
}
```

```
void son( ){
  while (true){
    wait (orange);
    wait (mutex);
    plate.remove (orange);
    signal (empty);
    signal (mutex);
  }
}
void daughter( ){
  while (true){
    wait (apple);
    wait (mutex);
    plate.remove(apple);
    signal (empty);
    signal(mutex);
  }
}
```

4.
```
semaphore empty1 = 1;
semaphore full1 = 0;
semaphore empty2 = 1;
semaphore full2 = 0;
void PA( ){
  while (true){
    record = read_from_disk( );
    wait (empty1);
    buffer1 = record;
    signal (full1);
  }
}
void PB( ){
  while (true){
    wait (full1);
    record = buffer1;
    signal (empty1);
    wait (empty2);
    buffer2 = record;
    signal (full2);
  }
}
```

```
void PC( ){
  while (true){
    wait(full2);
    print (buffer2);
    signal (empty2);
  }
}
```

		A	B	C	D	E	平均
	完成时间	3	15	8	20	10	
SJF	周转时间	3	13	4	14	2	7.2
(抢占)	带权周转时间	1.00	2.16	1.000	2.80	1.00	1.59
HRRN	完成时间	3	9	13	20	15	
	周转时间	3	7	9	14	7	8
	带权周转时间	1.00	1.17	2.25	2.80	3.50	2.14
RR	完成时间	4	18	17	20	15	
(q=1)	周转时间	4	16	13	14	7	10.8
	带权周转时间	1.33	2.67	3.25	2.80	3.50	2.71

2、A松弛度 $20ms-0ms-10ms=10ms$

B $50-0-10=40ms$

C $50-0-15=35ms$

∵ A<C<B

∴ A先执行任务，10ms后

B松弛度 $50-10-10=30ms$

C $50-10-15=25ms$

∵ C<B

∴ C执行，25ms后

A松弛度 $20-20-10=-10ms$

B $50-35-10=5ms$

执行任务B

顺序为 A→C→B

3、(1)、

	Allocation					work	need	Allocation	work+Allcation
	A	B	C	D		A B C D	A B C D	A B C D	A B C D
P0	0	0	3	2	P0	1 6 2 2	0 0 1 2	0 0 3 2	1 6 5 4
P1	1	0	0	0	P3	1 6 5 4	0 6 5 2	0 3 3 2	1 9 8 6
P2	1	3	5	4	P4	1 9 8 6	0 6 5 6	0 0 1 4	1 9 9 10
P3	0	3	3	2	P1	1 9 9 10	1 7 5 0	1 0 0 0	2 9 9 10
P4	0	0	1	4	P2	2 9 9 10	2 3 5 6	1 3 5 4	3 12 16 14

{P0, P3, P4, P1, P2}

(2)、① $Request_2(2,2,2) \le Need_2(2,3,5,6)$

② $Request_2(1,2,2,2) \le Available(1,6,2,2)$

③ 假设可分配P2

$Available=(0,4,0,0)$

$Allocation_2=(2,5,7,6)$

$Need_2=(1,1,3,4)$

④ 经安全性检查，$Need \le Available(0,4,0,0)$不成立

∴ 不能

(3)、系统立即满足P2请求(1,2,2,2)后，并未马上进入死锁状态。因为系统进程并未申请新的资源，并因得不到资源而阻塞，只有当进程提出新的请求，导致没有执行完的多个进程因得到资源而阻塞并形成循环等待链时，才发生死锁。

Python 实现 银行家算法

5. semaphore barber_ready=0;　　2023.03.16　　张益心
semaphore customer_ready=0;
semaphore payment=0;
semaphore receipt=0;
semaphore mutex=1;
int waiting=0;
void barber(){
while(true){
wait(customer_ready);
printf("理发师正在理发");
signal(payment);
wait(receipt);
signal(barber_ready);
}
}
void customer(){
wait(mutex);
if(waiting==N){
signal(mutex);
printf("顾客离开,沙发已满");
return;
}
waiting++;
signal(mutex);
signal(customer_ready);
wait(barber_ready);
wait(mutex);
waiting--;
signal(mutex);
wait(payment);
printf("顾客付费");
signal(receipt);
}

第三章

一.2. 对处理机进行分配

3. 指一个作业从提交给系统开始,到该作业完成并退出系统的总时间

4. 指两个或多个进程执行过程中,因互相等待对方持有的资源而无法继续执行的状态

三.2. 提交 后备 完成

6. 预防死锁 避免死锁 检测死锁 解除死锁

五.1.

		A	B	C	D	E	平均
FCFS	完成时间	3	9	13	18	20	
	周转时间	3	7	9	12	12	8.6
	带权周转时间	1.00	1.17	2.25	2.40	6.00	2.56
SJF	完成时间	3	9	15	20	11	
(非抢占)	周转时间	3	7	11	14	3	7.6
	带权周转时间	1.00	1.17	2.75	2.80	1.50	1.84